Economic Analysis of Software Defined Networking (SDN) under Various Network Failure Scenarios

Murat Karakus
Department of Computer and Information Science
Indiana University Purdue University Indianapolis
Indianapolis, IN 46202

Arjan Durresi
Department of Computer and Information Science
Indiana University Purdue University Indianapolis
Indianapolis, IN 46202

***Abstract*—Failures are inevitable in an operational network. They can happen anytime in different sizes and components of a network. They impact the network economics regarding CAPEX (Capital Expenditure), OPEX (Operational Expenditure), revenue lost due to service provisioning cut and so on. In order to mitigate the damages resulting from these failures, reactions of network architectures and designs are crucial for the future of the network. Recently, Software Defined Networking (SDN) has got the attention of researchers from both academia and industry as a means in order to increase network availability and reliability due to features, such as centralized automated control and global network view, it promises in networking. To this end, we explore the effects of programmable network architectures, i.e. SDN technology, and traditional network architectures, i.e. MPLS (Multiprotocol Label Switching) technology, on network economics by exploiting *Number of Satisfied Service Requests* and our predefined *Unit Service Cost Scalability* metrics under network failure scenarios: i) a random single data plane link failure and ii) a random controller (i.e. control plane) failure. To the best of our knowledge, this study is the first to consider a comparison of a programmable network architecture, i.e. SDN, along with different control plane models, Centralized (Single) Control Plane (CCP), Distributed (Flat) Control Plane (DCP), and Hierarchical Control Plane (HCP), and a non-programmable network architecture, i.e. MPLS, regarding network economics in case of network failures.**

***Index Terms*—SDN, Economics, Failure, Cost, CAPEX, OPEX**

I. Introduction

Availability in networks is one of the crucial attributes for the future of a network. When a failure happens in a network, it is important that service disruption for customers of the network are minimal because customers and the services are sources for the revenue/economics of the network. Therefore, detection of and recovery from a failure as quickly as possible has importance to mitigate the service and performance degradation in networks.

Failures are prevalent events that can occur independent of damage size, hardware, software, and so on in any network. Regardless of network type, size, and services, they can adversely affect the productivity, management, and service provisioning in a network. These damaging events, in turn, can result in an impact in CAPEX (Capital Expenditure), OPEX (Operational Expenditure), and revenue of a network. As a result, they can have a major financial impacts on service providers. According to [1], an hourly cost of downtime for computer networks is USD 42,000. A company, for example, suffering from an average downtime of 100 hours a year can lose more than $4 million per year. Also, the study in [2] states that cloud networks from 38 cloud providers amass losses estimated at USD 273 million and 1,600 hours of disruptions due to application and infrastructure failures.

Impacts of failures can be different depending on the architecture, programmable, e.g. Software Defined Networking (SDN) or traditional, e.g. MPLS (Multiprotocol Label Switching), as well as topology used in networks. In order to mitigate the damages resulting from these failures, reactions of network architectures and designs are crucial for the future of the network. Recently, SDN has got the attention of researchers from both academia and industry as a means in order to increase network availability, reliability, and revenue and as well as decrease network costs, thereby service costs, due to features, such as centralized automated control and global network view, it promises in networking.

In this paper, we investigate how programmable network architectures, i.e., SDN technology, affect the network economics compared to traditional network architectures, i.e., MPLS technology, in case of failures. Also, we explore the economic impact of failures in different SDN control plane models: Centralized (Single) Control Plane (CCP), Distributed (Flat) Control Plane (DCP), and Hierarchical Control Plane (HCP). We exploit our predefined metric called *Unit Service Cost Scalability* to evaluate economic performances of SDN architecture along with aforementioned control plane models and MPLS architecture under different failure scenarios. We consider two different failure types: i) a random single data plane link failure and ii) a random controller (i.e., control plane) failure. To the best of our knowledge, this paper is the first to consider a comparison of SDN architecture along with different control plane models and MPLS architecture regarding network economics under network failure scenarios. This work also aims at being a useful primer to providing insights regarding how network architectures and control plane models perform concerning network economics under failures for network owners to plan their investments accordingly.

In the rest of the paper, Section II gives a quick snapshot of the papers that study failure detection and recovery. In Section III, we explain our method along with experimental details and the metric used in economic analysis of the foregoing network architectures and control plane models in case of certain failure scenarios. While Section IV presents economic performances of both SDN models and MPLS in the data plane link failure scenario, Section V discusses economic performances of SDN models in the control plane (i.e. controller) failure scenario. We summarize the paper with concluding remarks in Section VI.

II. Related Work

Failure detection is the first step of dealing with a failure in a network. A simple and inefficient approach is to probe each switch on each link in the network utilizing some control

This is the author's manuscript of the article published in final edited form as:
Karakus, M., & Durresi, A. (2019). Economic Analysis of Software Defined Networking (SDN) Under Various Network Failure Scenarios. ICC 2019 - 2019 IEEE International Conference on Communications (ICC), 1–6. https://doi.org/10.1109/ICC.2019.8761124

messages such as hello messages, e.g., LLDP (Link Layer Discovery Protocol) [3]. However, this solution suffers from some problems such as imprecise detection of the failed device and scalability issues. Kozat et al. [4] propose a more scalable approach that revolves around the controller computing an Eulerian cycle across all links under its responsibility. Xu et al. [5] utilize Monitoring Flow Entries-based Link Failure Detection (MLFD) instead of LLDP-based failure detection. MLFD mechanism forms a monitoring tree path consisting of switches, which is probed using Link Monitoring (LM) packets. In [6], the authors propose an OpenFlow-like pipeline design called SPIDER that provides a detection mechanism based on switches' periodic link probing. Bidirectional Forwarding Detection (BFD) [7] with fast failover group type of OpenFlow is another approach exploited in failure detection.

After the detection of a failure, the network has to deal with recovery of paths (i.e., recomputation of new paths), that are broken down, in order to fulfill the flows that are affected by the failure. Failure recovery has two schemes: Protection and restoration. While protection involves a proactive behavior by installing backup paths before a failure occurs, the forwarding decisions are made after a failure happens in the restoration case. In [8], the authors exploit the fast failover group type and BFD to switch between two disjoint paths (working and protected) before a failure occurs in the network. Ramos et al. [9] utilize source routing to compute a secondary path for every path and storing it in the packet header along with the primary path. [10] introduces a framework, CORONET, which is a system for recovery from multiple link failures in data plane.

III. Economic Analysis of Network Failures

We study possible economic effects of different types of network failures in SDN networks and MPLS. We investigate two different failure scenarios: a random data plane link failure and a random control plane (i.e., controller) failure. These failure types are possible common failures in an SDN or MPLS networks. They also provide network administrators with insights to understand economic impact of failures in different domains in a network.

Carrier-grade networks require sub 50 ms failure recovery time not to cause a significant loss in service connectivity, customers subscriptions, and network revenue. Studies have shown that preserving such a fast recovery time is more possible with protection schemes without involving controllers in online decision-making process [8]. However, our goal is not to propose a new standalone failure recovery mechanism. We aim to economically analyze SDN architecture along with some popular control plane models used in SDN and MPLS architecture in case of different types of network failures scenarios. Therefore, we keep the following points in our mind (for all scenarios where applicable) while conducting this study in order to economically evaluate the preceding architectures and control plane models:

- We are not interested in or concerned with the speed of a detection/recovery mechanism since our goal is not to introduce a standalone network failure detection and recovery framework, which is out of this paper's scope. We exploit a failure detection/recovery mechanism for each failure scenario from the literature and use the same framework for all control plane models to obtain the start and finish times of failure detection/recovery while conducting the economic analysis.
- We concern about the economic impact of the failures in different control plane models. Therefore, using a framework with lower detection/recovery time affects economic values at the same ratio for all SDN models and MPLS network.
- Also, there is usually (a possibility for) another study introducing lower failure detection/recovery time. Thus, there is no end to find the best framework to name and use in a study similar to this.

Also, we exploit our previously proposed metric, *Unit Service Cost Scalability*, to evaluate economic performances of corresponding SDN control plane models and MPLS in the analysis. We present the metric shortly for readers in Subsection III-A For more details; we refer the readers to the study presented in [11].

A. Unit Service Cost Scalability Metric

One of the challenging questions about scalability is that whether the cost of the system to provide a service has any impact on scalability of the system. Jogalekar and Woodside [12] present that increased resource capacity in a system should be in proportion to the cost of the system while maintaining the quality of service. Also, in [13], Total Cost of Ownership (TCO) is stated as one of attributes that need to be considered while discussing network scalability. In addition, [14] proposes a scalability metric, called "P-Scalability", taking into account the cost of the system in distributed systems.

In this context, we define a metric called *Unit Service Cost Scalability* to evaluate unit service cost performance of a network for service requests. It considers the network workload and costs incurred to maintain the QoS at the same level for all service requests in the network.

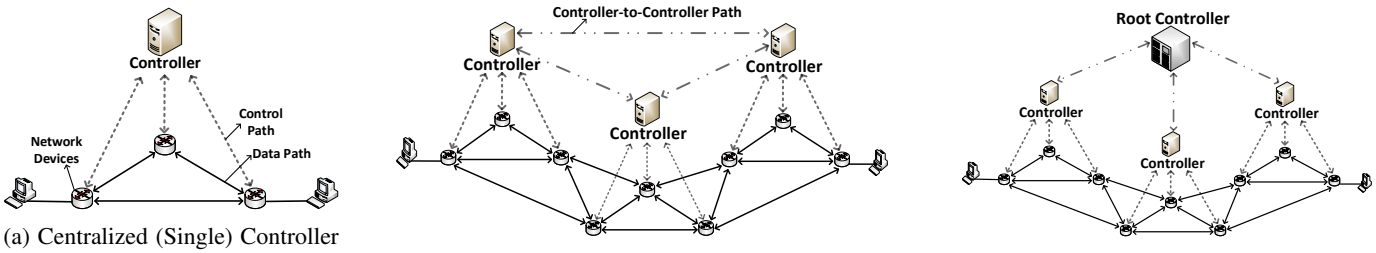
We characterize the unit cost for a service (request) from a service tier as a function of network CAPEX, OPEX, and Workload over a certain time period. We refer Workload to service requests of all service tiers coming from users/customers to and *satisfied* by the network. The general unit cost framework for a service (request) with one QoS parameter (bandwidth) from a service tier is shown in Eq. 1. This formula implies that the unit service cost for a request from certain service tier is the ratio of TCO (CAPEX + OPEX) over workload in a given period.

$$\mathcal{C}_{bw_j} = f(\mathbb{C}, \mathbb{O}, W) = \begin{cases} \frac{\mathbb{C}+\mathbb{O}}{\sum_{j=1} w_j \cdot |bw_j|} \cdot |bw_j| & \text{before } \delta \\ \frac{\mathbb{C}+\mathbb{C}_\delta+\mathbb{O}+\mathbb{O}_\delta}{\sum_{j=1} (w_j+w_{\delta_j}) \cdot |bw_j|} \cdot |bw_j| & \text{after } \delta \end{cases} \tag{1}$$

where bw_j, $|bw_j|$, $\mathcal{C}_{bw_j}$, $\mathbb{C}$, $\mathbb{O}$ represent the type of (i.e. bandwidth) service with tier j, the numerical value of the service tier bw_j, the unit cost of the service bw_j, CAPEX, and OPEX in a time period (e.g. month, year), respectively. w_j and w_{δ_j} represent the workload and possible additional workload of service bw_j and $W = \sum_{j=1} w_j$ and $W_\delta = \sum_{j=1} w_{\delta_j}$. $\mathbb{C}_\delta$, $\mathbb{O}_\delta$, and W_δ represent possible extra CAPEX, OPEX, and workload, respectively, incurred after introducing different kinds of changes/upgrades (represented as δ) in the network.

B. Experimental Setup

We have considered CCP, DCP, and HCP (shown in Fig. 1) as the SDN control plane models in this study while conduct-



(a) Centralized (Single) Controller Design (CCP)

(b) Distributed (Flat) Controller Design (DCP)

(c) Hierarchical Controller Design (HCP)

Fig. 1: A representational overview of popular SDN control plane models. The two-sided solid, dashed, and dashed-dotted arrows represent two-way data path among network devices, control path between controller and data plane devices, and controller-to-controller path among controllers, respectively. In 1a (CCP), there is one main controller with global network state. In 1b (DCP), each controller manages a sub-network/domain of the whole network and has its own local network view, which is, in turn, abstracted as a logical node to its neighboring controllers. In 1c (HCP), there are layers where local controllers with local view reside and are responsible for different sites (sub-domains) and a Root controller on top with global network view is responsible for global applications like routing. We note that the data plane and control plane topologies shown in this figure are just representative purposes.

ing analysis to evaluate their impact on network economics in failure scenarios.

We have used Mininet emulator with POX controller in SDN models. While there is one controller in CCP model, we have divided the whole network into 4 fully-connected sub-networks with a primary controller for each in control planes of DCP and HCP models and provided 16 switches and 24 links in data planes of all SDN models and MPLS. The data plane topology is the same in all SDN models and MPLS cases. There is also a Root controller on top of local domain controllers in HCP model. Regarding MPLS setting, we have used ns3 network simulator. We needed to use a signaling protocol such as RSVP-TE (Resource Reservation Protocol-Traffic Engineering) and/or CR-LDP (Constraint-based Label Distribution Protocol) to support constraint-based routing in MPLS. Since none of them has been implemented in ns3 at the time of this writing and it is time-consuming and effort-greedy to implement them in ns3, we have generated extra packets between network elements to mimic link state advertisements and state refresh messages for LSPs from aforementioned signaling protocols in MPLS.

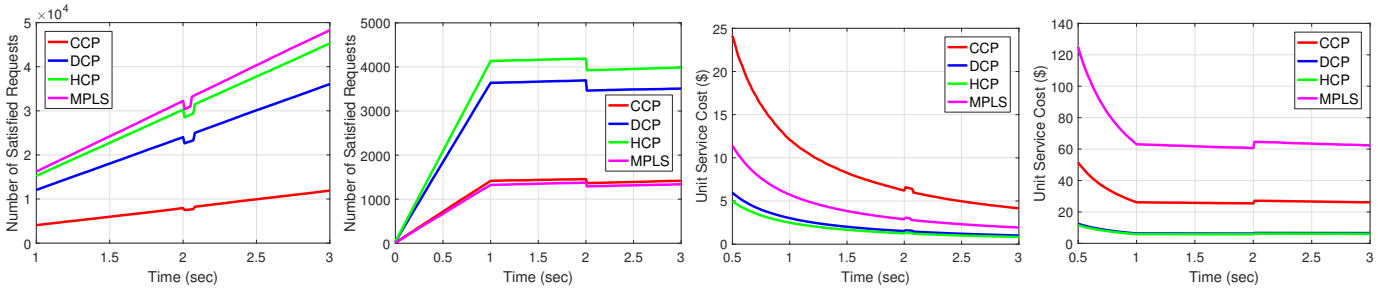
In the experiments, we have used 1 Mbps flow sending rate for all service requests. Also, we have used a modified version of Waxman [15] random topology generator defined by Erdos-Renyi random graph model to randomly create the networks while preserving connectivity degrees of nodes (i.e. switches) as three in all models. Furthermore, we have conducted a heuristics, i.e. A*Prune Algorithm [16], to find a feasible path through the network. In all scenarios, in the first part of the experiments, we have provided enough bandwidth (100 Gbps) in links so that there is no service request rejection due to network resource limitations, while, in the second part of the experiments, we have reduced the link bandwidth to 1 Gbps to see their performances under network resource limitations. In the experiments, the corresponding failure type occurs at the 2nd second in both failure scenarios. Moreover, we have averaged 20 runs for each experiment to achieve and exceed 95% statistical significance in all scenarios. Finally, all experiments were performed on Ubuntu 14.04 in Oracle VirtualBox using an Intel Core i7-5500 system with 12GB RAM.

IV. Scenario 1: Data Plane (Link) Failure

Data plane (link) failure is one of the failure types that we have investigated in our analysis for economic analysis of failures in different SDN control plane models and MPLS. There exist many proposals for link failure detection/recovery methods in a network. Some of the prevalent methods exploited for failure detection are BFD sessions, Loss of Signal (LOS), and LLDP packets. In this scenario, we only consider a random single link failure.

For the link failure scenarios, we have utilized the methods proposed in [17]. This study implements two well-known mechanisms of failure recovery, i.e., protection and restoration, in OpenFlow networks. In the case of protection, the alternative paths are reserved before the failure occurs in the network. In the case of restoration, alternative paths are not established until a failure occurs. The controller in restoration must notify all the affected switches about a recovery action immediately. For implementation of the protection scheme, the "Group Table" concept specified for OpenFlow in its version 1.1 is used. OpenFlow introduces the fast-failover group type in order to perform fast failover without needing to involve the controller. Any group entry of this type consists of two or more action buckets with a well-defined order. The status of the bucket can be changed by the monitored port going into the "down" state or through other mechanisms such as BFD. In the study, BFD was used to detect the failures. Once BFD declares the failure in the working link, the action bucket associated with this link in the group table is made unavailable by changing the value of the alive status. For the restoration method, once the failure has been detected, the controller is notified about it in order to calculate new paths for the affected flows. The controller takes then the necessary actions such as path recomputation for flows affected by the failure and installation of new flow rules for newly computed paths in the corresponding switches over the new paths.

As to the MPLS case, we have used a similar method to the one explained above for the SDN models. In the protection case, we have utilized a link protection approach by pre-programming next-hop port values into the router FIB awaiting activation, which happens in milliseconds following the failure detection. In the restoration case, on the other hand, the head

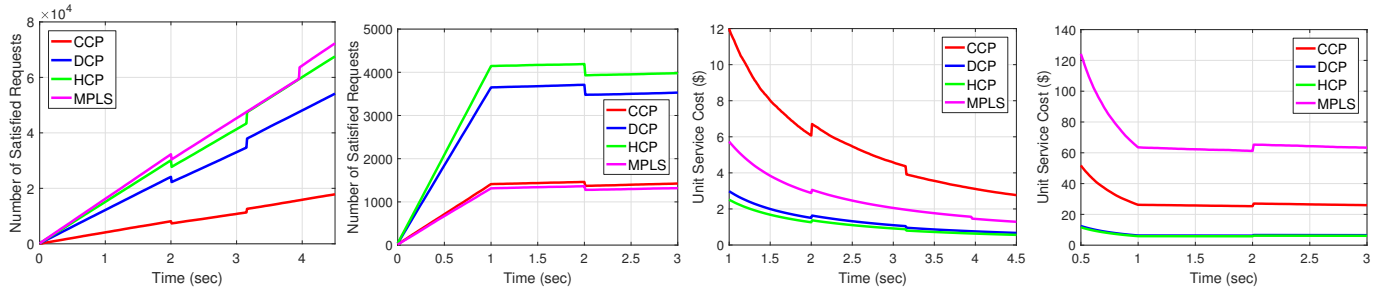


(a) Total number of satisfied requests vs. Time - 100 Gbps link bandwidth - Protection case

(b) Total number of satisfied requests vs. Time - 1 Gbps link bandwidth - Protection case

(c) Unit service cost vs. Time - 100 Gbps link bandwidth - Protection case

(d) Unit service cost vs. Time - 1 Gbps link bandwidth - Protection case



(e) Total number of satisfied requests vs. Time - 100 Gbps link bandwidth - Restoration case

(f) Total number of satisfied requests vs. Time - 1 Gbps link bandwidth - Restoration case

(g) Unit service cost vs. Time - 100 Gbps link bandwidth - Restoration case

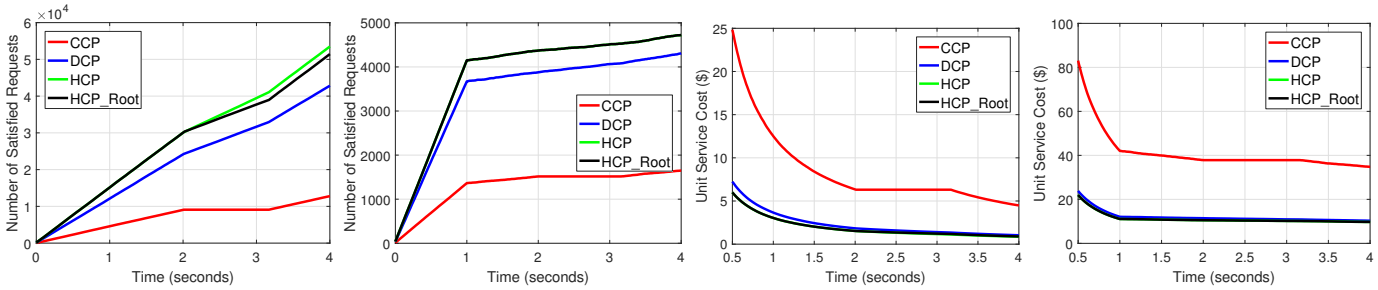
(h) Unit service cost vs. Time - 1 Gbps link bandwidth - Restoration case

Fig. 2: Total number of satisfied requests and respective unit service cost performances of networks in case of data plane (link) failure with use of protection and restoration schemes under 100 Gbps and 1 Gbps link bandwidth cases.

end of each path for each flow on the failed links recomputes a new path following the failure detection. Regarding the failure detection and recovery activation, BFD sessions were established between routers for each link in the networks.

Fig. 2 shows the total number of satisfied requests and unit service cost performances in case of a random single data plane link failure with use of protection (Fig. 2a, 2b, 2c, 2d) and restoration (Fig. 2e, 2f, 2g, 2h) schemes under 100 Gbps and 1 Gbps link bandwidth cases. We have used the protection and restoration schemes explained earlier for link failure detection and recovery. In 100 Gbps link bandwidth case, the total number of satisfied requests in the network increases in both protection (Fig. 2a) and restoration (Fig. 2e) cases for all SDN control plane models and MPLS until the failure happens because there are enough bandwidth to use in the links. Once the failure happens, the number of satisfied flows in the network shows a reduction because the flows served over the failed link are not satisfied anymore. The reduction is the lowest in CCP and highest in HCP (465 in CCP, 1392 in DCP, 1750 in HCP, and 1862 in MPLS) among all SDN models and MPLS in both protection and restoration cases owing to the total number of satisfied flows in the networks. However, the total number of satisfied requests for all models continue to increase immediately after the reduction because there are enough bandwidth in the links for the upcoming requests. On the other hand, in 1 Gbps link bandwidth case, the number of satisfied requests in the network increase until links become loaded in both protection (Fig. 2b) and restoration (Fig. 2f) cases for all SDN control plane models and MPLS. After links become loaded, it shows a steady-like behavior until the failure. Once the failure happens, it shows the lowest reduction in CCP while the highest is in HCP (91 in CCP, 229 in DCP, 261 in HCP, 84 in MPLS) as in the previous link bandwidth case in both protection and restoration schemes. The total number of satisfied requests for all models stay steady, unlike the 100 Gbps link bandwidth case, during the recovery phase because new flows cannot be satisfied in the network since the links are loaded in both SDN models and MPLS. However, the recovery phase cannot be completed in 1 Gbps link bandwidth case because the flows over the failed link cannot be rerouted from failed link to the other links because all links are loaded/full and cannot handle more flows. Therefore, the number of flows do not increase again after the failure, unlike the 100 Gbps link bandwidth case.

Regarding the unit service cost, in 100 Gbps link bandwidth case, the results reveal that it decreases as the number of satisfied flows increase until the failure occurs for all SDN models and MPLS in both protection (Fig. 2c) and restoration (Fig. 2g) cases. While the unit service cost increase ratio is the highest in CCP ($\sim$6.1% in CCP, $\sim$3.1% in DCP, $\sim$1.9% in HCP, and $\sim$4.6% in MPLS), HCP shows the lowest increase ratio among all models in protection and restoration schemes. After the recovery is complete, the unit service cost starts showing a reduction again in both protection and restoration schemes. In 1 Gbps case, the unit service cost decreases as the total number of satisfied requests in the network increase until the links become loaded in both protection (Fig. 2d) and restoration (Fig. 2h) cases. Then, it reflects a steady-like behavior until the failure happens. The link failure results in a sudden increase ($\sim$5.7% in CCP, $\sim$2.2% in DCP, $\sim$1.6% in HCP, and $\sim$6.5% in MPLS) in protection and restoration schemes in the unit service cost as in the 100 Gbps link bandwidth case. However, the unit service cost do not show a reduction again in either of protection and restorations



(a) Total number of satisfied requests vs. Time - 100 Gbps link bandwidth

(b) Total number of satisfied requests vs. Time - 1 Gbps link bandwidth

(c) Unit service cost vs. Time - 100 Gbps link bandwidth

(d) Unit service cost vs. Time - 1 Gbps link bandwidth

Fig. 3: Total number of satisfied requests and respective unit service cost performances of networks in case of controller failure under 100 Gbps and 1 Gbps link bandwidth cases.

schemes unlike the 100 Gbps link bandwidth case since the recovery cannot be completed due to the loaded links.

V. Scenario 2: Control Plane (Controller) Failure

As SDN brings many advantages to networking, it also suffers from various problems. One of the serious problems of SDN is that the controller may be a critical point of failure, which can result in an overall network unavailability. Therefore, the design of a fault-tolerant control plane is a must for a SDN-based network. There might be varying number of reasons for failure of a controller: hardware failure (e.g., controller server hardware), software failure/bug in the server operating system and controller software, power outages and so on. One basic solution for a controller failure is use of redundant controller(s) (i.e., backup/standby controller) in order to automatically take critical responsibilities over network infrastructure control and data flows management from the failed primary controller in case of a controller failure. While this procedure is called controller failover, the reverse of the procedure (i.e., restoring the primary controller) is called controller failback. As of OpenFlow protocol version 1.2.0, it provides the possibility to configure one or more backup controllers which can assume the network control in case of failure using controller role change mechanism, but OpenFlow does not provide any coordination mechanism between the primary and the backup controllers. Therefore, it is network administrators' responsibility to provide such a synchronization method for consistency among them to handle controller failure and add resiliency without happening any fatal damage to network services and customer satisfaction.

In the control plane failure case, we have utilized the method described in [18]. In this scenario, every network domain is managed by a single controller, and another controller is used as backup for every domain controller that can take over its role in case the primary fails. In case a failure occurs, and to ensure a smooth transition to a new primary, in this particular instance of a fault-tolerant architecture, the controller store the network and application related state in a shared data store.

Fig. 3 shows the total number of satisfied requests and respective unit service cost performances of networks in case of a controller failure in SDN models (CCP, DCP, and HCP) under 100 Gbps and 1 Gbps link bandwidth cases. We also evaluated Root controller failure case in HCP model. In 100 Gbps link bandwidth case (Fig. 3a), although the total number of satisfied requests in the networks show an increase all the time, a primary controller failure results in some reduction in increase ratio of the total number of satisfied requests in the network in DCP, HCP, HCP_Root controller failure cases. Also, since all inter-domain connection requests are affected, this reduction is more in case of HCP_Root controller failure. In CCP model, the total number of satisfied requests in the network stay the same once a controller failure happens because there is only one controller. However, the total number of satisfied requests in the network start increasing after the backup controller takes over the network management responsibilities. In 1 Gbps link bandwidth case (Fig. 3b), the total number satisfied requests in the network increase until the links become loaded and then shows a steady-like behavior in all control plane models. In CCP model, it does not increase in controller failure case since there is only one domain controlled by one controller although it shows a very little increase in DCP and HCP models because some requests may still find enough remaining bandwidth in the links depending on the source and destination pairs. Also, HCP and HCP_Root failure cases do not make any difference in results because the total number of satisfied requests in the network are the same for both failure cases. It is the same because the network becomes loaded before the controllers reach their requests handling capacities.

Regarding the unit service cost, it is shown that it decreases as the number of satisfied requests are increasing in both 100 Gbps (Fig. 3c) and 1 Gbps (Fig. 3d) link bandwidth cases. In Fig. 3c, the unit service cost stays steady in CCP model when the controller failure occurs because the satisfied requests number in the network do not increase while the unit service cost shows a continuous decreasing behavior in DCP and HCP models in that phase. In Fig. 3d, the unit service cost decreases fast until the links become loaded and then shows a slow reduction in all models. In both 100 and 1 Gbps link bandwidth cases, HCP_Root controller failure does not make a noticeable difference compared to a controller failure in HCP concerning the unit service cost.

Acknowledgement

This work was partially supported by the National Science Foundation under Grant No.1547411 and by the U.S. Department of Agriculture (USDA) National Institute of Food

and Agriculture (NIFA) (Award Number 2017-67003-26057) via an interagency partnership between USDA-NIFA and the National Science Foundation on the research program Innovations at the Nexus of Food, Energy and Water Systems.

VI. Conclusions

We have explored effects of programmable network architectures, i.e., SDN technology, and traditional network architectures, i.e., MPLS technology, on network economics by exploiting *Number of Satisfied Service Requests* and our predefined *Unit Service Cost Scalability* metrics under network failure scenarios: i) a random single data plane link failure and ii) a random controller (i.e. control plane) failure. The experiments have revealed that the unit service cost shows an increase in case of data link failure in all SDN models and MPLS. While CCP model shows the highest increase ratio, HCP shows the least increase ratio among all SDN models and MPLS in 100 Gbps link bandwidth case. Also, this increase ratio is the highest in MPLS case in 1 Gbps link case. In control plane (i.e. controller) failure scenario, the unit service cost stays steady in CCP model once the controller failure occurs because the satisfied requests number in the network do not increase while the unit service cost shows a continuous decreasing behavior in DCP and HCP models in that phase in case of 100 Gbps link bandwidth case. Also, the unit service cost decreases fast until the links become loaded and then shows a slow reduction in all models in case of 1 Gbps link bandwidth case. Moreover, in both 100 and 1 Gbps link bandwidth cases, HCP_Root controller failure does not make a noticeable difference concerning the unit service cost compared to a controller failure in HCP.

References

[1] "Q&A: How Much Does an Hour of Downtime Cost?" Gartner, Tech. Rep., September 20009.

[2] C. Cérin, C. Coti, P. Delort, F. Diaz, M. Gagnaire, Q. Gaumer, N. Guillaume, J. Lous, S. Lubiarz, J. Raffaelli *et al.*, "Downtime statistics of current cloud solutions," *International Working Group on Cloud Computing Resiliency, Tech. Rep*, 2013.

[3] "Ieee standard for local and metropolitan area networks - station and media access control connectivity discovery," *IEEE Std 802.1AB-2016 (Revision of IEEE Std 802.1AB-2009)*, pp. 1–146, March 2016.

[4] U. C. Kozat, G. Liang, and K. Kkten, "On diagnosis of forwarding plane via static forwarding rules in software defined networks," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, April 2014, pp. 1716–1724.

[5] H. Xu, L. Yan, H. Xing, Y. Cui, and S. Li, "Link failure detection in software defined networks: an active feedback mechanism," *Electronics Letters*, vol. 53, no. 11, pp. 722–724, 2017.

[6] C. Cascone, D. Sanvito, L. Pollini, A. Capone, and B. Sansò, "Fast failure detection and recovery in sdn with stateful data plane," *International Journal of Network Management*, vol. 27, no. 2, 2017.

[7] D. Katz and D. Ward, "RFC 5880: Bidirectional Forwarding Detection (BFD)," 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5880.txt

[8] S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester, "A demonstration of fast failure recovery in software defined networking," in *International Conference on Testbeds and Research Infrastructures*. Springer, 2012, pp. 411–414.

[9] R. M. Ramos, M. Martinello, and C. E. Rothenberg, "Data center fault-tolerant routing and forwarding: An approach based on encoded paths," in *2013 Sixth Latin-American Symposium on Dependable Computing*, April 2013, pp. 104–113.

[10] H. Kim, M. Schlansker, J. R. Santos, J. Tourrilhes, Y. Turner, and N. Feamster, "Coronet: Fault tolerance for software defined networks," in *2012 20th IEEE International Conference on Network Protocols (ICNP)*, Oct 2012, pp. 1–2.

[11] M. Karakus and A. Durresi, "Economic viability of software defined networking (sdn)," *Computer Networks*, vol. 135, pp. 81 – 95, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128618300823

[12] P. Jogalekar and M. Woodside, "Evaluating the scalability of distributed systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, no. 6, Jun. 2000.

[13] M. Behringer, A. Retana, R. White, and G. Huston, "RFC 7980: A Framework for Defining Network Complexity."

[14] P. Jogalekar and C. Woodside, "A scalability metric for distributed computing applications in telecommunications," *Teletraffic Science and Engineering*, vol. 2, pp. 101–110, 1997.

[15] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, Dec 1988.

[16] G. Liu and K. G. Ramakrishnan, "A*prune: an algorithm for finding k shortest paths subject to multiple constraints," in *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*, vol. 2, 2001, pp. 743–749 vol.2.

[17] "Openflow: Meeting carrier-grade recovery requirements," *Computer Communications*, vol. 36, no. 6, pp. 656 – 665, 2013.

[18] F. Botelho, A. Bessani, F. Ramos, and P. Ferreira, "On the design of practical fault-tolerant sdn controllers," in *Software Defined Networks (EWSDN), 2014 Third European Workshop on*, Sept 2014, pp. 73–78.